DOCUMENT RESUME

ED 152 277 IR 005 736

AUTHOR Brown, John Seely; Burton, Richard R.
TITLE A Paradigmatic Example of an Artificially Intelligent Instructional System.
INSTITUTION Bolt, Beranek and Newman, Inc., Cambridge, Mass.
SPONS AGENCY Advanced Research Projects Agency (DOD), Washington, D.C.
PUB DATE Jun 77
CONTRACT MDA903-76-C-0108
NOTE 27p.

EDRS PRICE MF-$0.83 HC-$2.06 Plus Postage.
DESCRIPTORS *Artificial Intelligence; *Computer Assisted Instruction; Conceptual Schemes; Feedback; *Instructional Systems; *Man Machine Systems; Mathematical Models; *Mathematics Instruction; On Line Systems; *Tutoring

ABSTRACT

This paper describes the philosophy of intelligent instructional systems and presents an example of one such system in the domain of manipulative mathematics--BLOCKS. The notion of BLOCKS as a paradigmatic system is explicated from both the system development and educational viewpoints. From a developmental point of view, the modular design of BLOCKS provides a working framework within which to explore different monitoring functions and various tutoring strategies. From an educational viewpoint, BLOCKS provides a dramatic example of the potential of a computerized intelligent tutor in a laboratory environment. By monitoring the student's behavior, the system can notice interesting situations and direct the student's attention to them. In this way, the computer can provide conceptual structure and guidance to a student's otherwise undirected experiences. (Author/VT)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

To appear in:

Proceedings of the First International Conference on Applied General Systems Research: Recent Developments --- Trends, Binghamton, New York, August 1977.

U.S. DEPARTMENT OF HEALTH, EDUCATION & WELFARE
NATIONAL INSTITUTE OF EDUCATION

THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL NATIONAL INSTITUTE OF EDUCATION POSITION OR POLICY

A PARADIGMATIC EXAMPLE OF AN ARTIFICIALLY INTELLIGENT INSTRUCTIONAL SYSTEM

John Seely Brown and Richard R. Burton
Bolt Beranek and Newman Inc.
Cambridge, Mass.

June 1977

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY

John Seely Brown

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC) AND USERS OF THE ERIC SYSTEM."

ED152277

IR005736

Acknowledgements

We are especially indebted to Dr. Guy Groen whose suggestion led us to explore the use of production rules as a representation for student modelling and who coined the term "paradigmatic system." We would also like to express our thanks to Robert Weissman for developing the RITA rules for BLOCKS, to Kathy Larkin for her suggestions and assistance on this paper, and to Mark Spikell who suggested we consider the domain of attribute blocks for developing our intelligent tutoring paradigm.

This research was supported by the Advanced Research Projects Agency, Air Force Human Resources Laboratory, Army Research Institute for Behavioral and Social Sciences, and Navy Personnel Research and Development Center, under Contract No. MDA903-76-C-0108.

ABSTRACT

This paper describes the philosophy of intelligent instructional systems and presents an example of one such system in the domain of manipulative mathematics--BLOCKS. The notion of BLOCKS as a paradigmatic system is explicated from both the system development and educational viewpoints. From a developmental point of view, the modular design of BLOCKS provides a working framework within which to explore different monitoring functions and various tutoring strategies. A particularly interesting discovery that arises from experiences with the system is the need for a structural model of a student to direct the content and level of the tutor's comments. From an educational viewpoint, BLOCKS provides a dramatic example of the potential of a computerized intelligent tutor in a laboratory environment. By monitoring the student's behavior, the system can notice interesting situations and direct the student's attention to them. In this way, the computer can provide conceptual structure and guidance to a student's otherwise undirected experiences.

TABLE OF CONTENTS

	Page
A Paradigmatic Example	2
Description of the Game	3
Protocols	4
Rules in the Attribute Blocks Domain	11
Rule Application	13
System Description	14
Executive	15
Natural Language Understander	15
Environment Maintenance	15
Monitors	16
Remaining Possibilities monitor	16
Information Gain monitor	16
The Tutor and its Tutoring Strategies	17
Expert Rule-based Deducer	17
Conclusions	18
References	19

A PARADIGMATIC EXAMPLE OF AN ARTIFICIALLY INTELLIGENT INSTRUCTIONAL SYSTEM

John Seely Brown and Richard R. Burton
Bolt Beranek and Newman Inc.
Cambridge, Mass.

This is a period of dramatic advances in computer technology which should change the way computers are employed in instruction. Technological advances will decrease the cost of computer hardware to the extent that each student will have available computational resources which are currently restricted to a few elite users. Traditional computer-assisted instruction (CAI) paradigms were developed under the assumption that computational power is a scarce resource, and these paradigms are, for the most part, incapable of exploiting the latest technological advances. To effectively use the increased availability of computational power requires a re-evaluation of the role of the computer in instructional paradigms.

This paper describes research directed at understanding and designing artificially intelligent instructional systems -- based on AI paradigms -- which take fuller advantage of increased computational power. The kind of instructional system we are investigating does more than spew forth its knowledge as factual information. It uses its knowledge base and problem-solving expertise to aid the student in several ways. First, it answers his questions and can evaluate his theories as well as critique his solution paths. Second, it can form structural models of his reasoning strategies. These structural models are used both to identify his fundamental misconceptions and to determine when and how to provide remediation, heuristic recommendations ("hints"), or further instruction.

In addition to the assumed computational power, the AI instructional paradigm differs from classical CAI in terms of the type of knowledge it seeks to develop. We are not focusing on techniques for teaching factual, textbook knowledge (which can often be competently handled by frame-oriented CAI or CMI systems). Instead, we are focusing on techniques for teaching procedural knowledge and reasoning strategies which are best learned through hands-on laboratory or problem-solving tasks, during which the student gets a chance to exercise his knowledge under the watchful and critical eye of an automated intelligent tutor. That is, an intelligent

instructional system attempts to mimic the capabilities of a laboratory instructor working on a one-to-one basis with a student, carefully diagnosing what the student knows, how he reasons and what kinds of deficiencies exist in his ability to apply his factual knowledge. The system then uses this inferred knowledge of the student to determine how best to critique and/or kibitz with him.

While we are still a long way from attaining this goal, we have developed an organization for intelligent instructional systems, (described in Brown [1975]) which appears fruitful. Our methodology for developing this organization (and the theory underlying it) has been to explore parts of the overall organization in "paradigmatic" systems. A paradigmatic system is an easily modified prototype system constructed over a carefully chosen domain of knowledge. This methodology allows experimentation with some aspect of the overall system by simplifying other aspects. In this paper we describe one of these paradigmatic systems -- BLOCKS. We have developed systems for other domains including electronic troubleshooting -- SOPHIE [Brown Burton and Bell 1975; Brown et al. 1976]; arithmetic drill and practice -- WEST [Burton and Brown 1976]; elementary algebra [Brown et al. 1975]; and procedural skills in arithmetic -- BUGGY [Brown et al. 1977]. In addition, systems of similar spirit are being developed by Goldstein [Goldstein 1976; Carr and Goldstein 1977].

A Paradigmatic Example

A prerequisite for a paradigmatic system is a subject domain that has a simple and elegant structure. The domain must have a logical formulation that is both well-defined and easily specifiable. In addition, the logical structures of the domain must support natural mappings (analogies) into the kinds of complex and real world domains that instructional systems are intended to handle.

A domain that appears to be ideal for this purpose derives from that part of the world of manipulatory mathematics known as attribute blocks [Greenes et al. 1972; Dienes and Golding 1966; Elementary Science Study 1968]. Although attribute blocks can be used to explore a rich variety of interesting, common-sense reasoning principles, we shall focus on just one application -- a game which combines the notions of logic, decision-making

and hypothesis formation into an interesting exercise on how to ask optimal questions and how to draw inferences from the answers.

Description of the Game

This game is played with the 32 attribute blocks, a deck of attribute cards and 2 looped strings. Each block has three attributes:

SIZE: small or large
COLOR: red or yellow or green or blue
SHAPE: triangle or square or circle or diamond

There is one block in the set of 32 for each possible combination of the values of the three attributes.

The deck is made up of 18 cards. Written on each card is an attribute value or the negation of a value.

1. LARGE	7. TRIANGLE	13. NOT YELLOW
2. SMALL	8. CIRCLE	14. NOT GREEN
3. BLUE	9. SQUARE	15. NOT TRIANGLE
4. RED	10. DIAMOND	16. NOT CIRCLE
5. YELLOW	11. NOT BLUE	17. NOT SQUARE
6. GREEN	12. NOT RED	18. NOT DIAMOND

The student takes the two looped strings and overlaps them as in Figure 1. This arrangement of the looped strings creates four areas. Area 1 is inside the string on the left (loop A) and outside the string on the right (loop B). Area 2 is inside loop B and outside loop A. Area 3 is inside both loops. Area 4 is outside both loops.

The squares labelled Card A and Card B in Areas 1 and 2 represent two cards which the teacher chooses from the deck of cards. The student is NOT told which cards have been chosen. The object of the game is for the student to guess the attribute value written on each of these two cards. To do this the student chooses blocks one at a time and asks the teacher where the block goes (according to the rule that a block is placed inside of loop A only if it satisfies the value on Card A, and inside of loop B only if it satisfies the value on Card B, e.g. if Card A=SQUARE and Card B=NOT BLUE, then the Large Yellow Square goes in Area 3). The student continues choosing blocks, asking where they go, and placing them there, until he believes he has placed enough blocks to uniquely identify (i.e. deduce) what each of the cards is. (The reader is encouraged to play this game using paper and pencil to get a feeling for the types of deductions involved.)

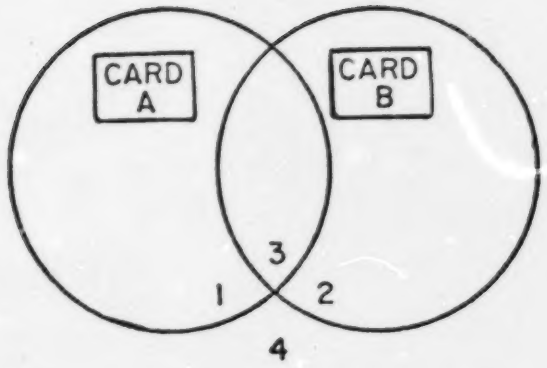


Figure 1

what can a computer do for this environment?

Manipulatory math tools represent, in our opinion, one of the best uses of simple, inexpensive technology. It was, therefore, with some trepidation that we considered contaminating this otherwise simple domain with "high technology". However, after watching numerous people use attribute blocks, we felt there were many important tasks that could be better accomplished (and in some cases, only accomplished) by having a knowledge-based CAI system. In particular, it was clear that questions like "When have I placed enough blocks?" or "What is the best next block to place?" requires concentration and training beyond that of most teachers. Indeed, that these are profitable questions to ask is seldom appreciated. The use and import of these questions will be seen in the following protocols.

Protocols

In this section, we present three annotated protocols of a hypothetical student using BLOCKS. Each protocol builds on the prior one and illustrates additional tutoring features which are realized by having the instructional system take on additional information processing capabilities. One of the interesting aspects of BLOCKS is the way that the domain of attribute blocks gains depth by the addition of these capabilities.

The annotations in these protocols include references to components in the basic architecture of BLOCKS. Figure 2 illustrates its functional decomposition into the modules referred to below. The first protocol stems from a relatively simple version of Figure 2 in which there are three monitors and a tutor. The first monitor (which heavily utilizes the expert) evaluates the student's conjecture about what a card is and determines into which of the following three categories the conjecture falls: 1) the conjecture is necessarily correct (i.e. the current block placements entail that card and only that card); 2) it is consistent with the known information (i.e. block placements) but there are still other possibilities for the card; or 3) the conjecture is inconsistent with the current block placements. If the conjecture is inconsistent, this monitor also selects a counter-example. The second monitor determines if the

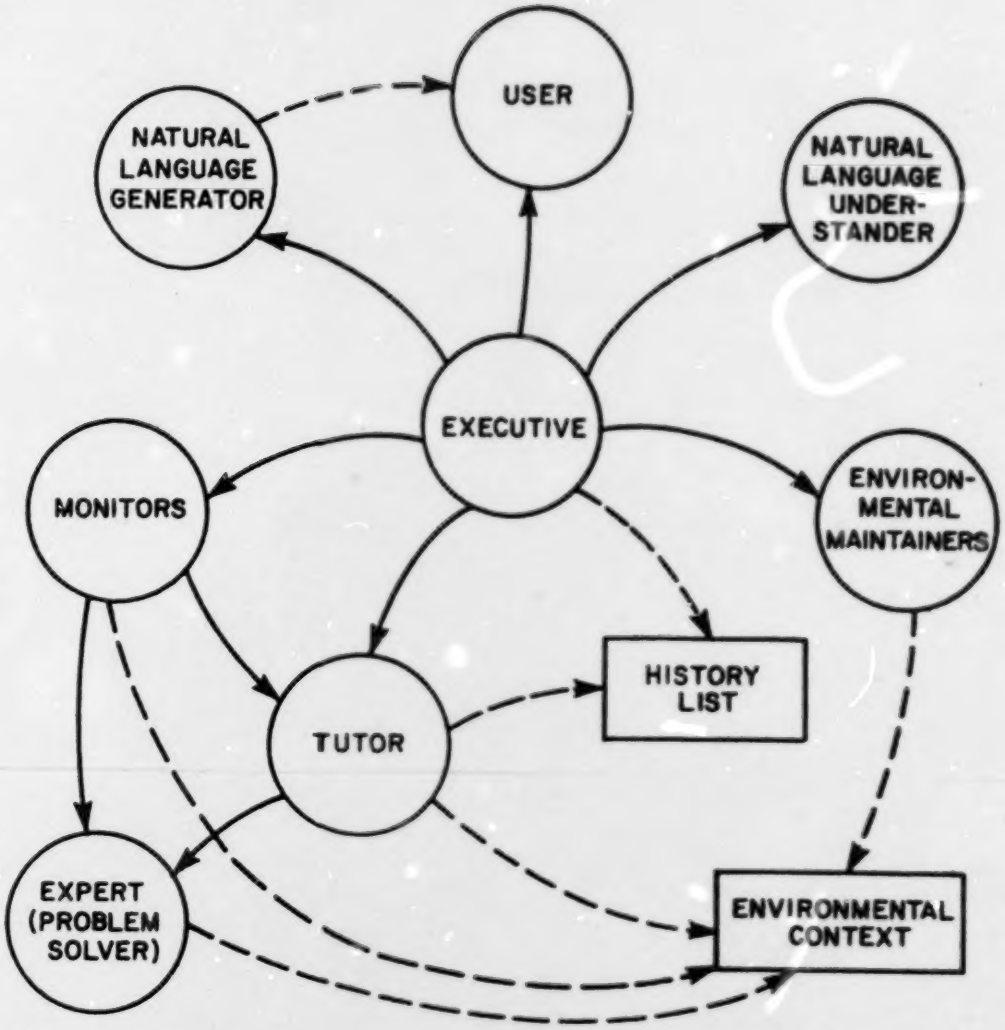

USER
NATURAL LANGUAGE GENERATOR
NATURAL LANGUAGE UNDER-STANDER
EXECUTIVE
MONITORS
ENVIRON-MENTAL MAINTAINERS
HISTORY LIST
TUTOR
EXPERT (PROBLEM SOLVER)
ENVIRONMENTAL CONTEXT

Figure 2

placement of a chosen block could have been deduced from the existing blocks, i.e. if it is a logically redundant block. From a strict information theoretic point of view, such choices reflect informationless questions. Detecting such questions is an easy task for the monitor to perform, but a difficult one for a human teacher. The third monitor determines when there is sufficient information (blocks already placed) to deduce what one of the cards is.

These monitors are constantly watching the environment (i.e. student moves). They "wake up" the tutor whenever they detect something of interest, informing the alerted tutor of what it was that they found worth waking him up for. The tutor invokes one of several Socratic teaching strategies for trying to get the student to realize concepts such as: i) the notion of having contradictory evidence to a hypothesis; ii) the notion of having sufficient information to make a deduction; or iii) the concept of redundancy and why his particular block choice was redundant.

PROTOCOL 1

The lines with "***" in the left column were typed by the student at his own initiative. The lines with ">>>" in the left column were typed by the student in response to a query by the tutor.

```
WELCOME TO THE ATTRIBUTE BLOCKS WORLD

TWO CARDS HAVE BEEN CHOSEN.
```

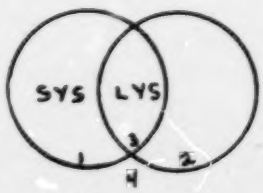
System selects two cards.

```
----CHOOSE A BLOCK----
***LARGE Y SQ
```

The student tells the system which block he would like placed. The system recognizes abbreviations of block descriptions.

```
PUT THE LARGE YELLOW SQUARE IN AREA 3.
```

System tells him where to place his block in accordance with what the cards are.




```
***SMALL YELLOW SQUARE
PUT THE SMALL YELLOW SQUARE IN AREA 1.
YOU SHOULD BE ABLE TO DEDUCE CARD B NOW.
```

The monitor watching the possibilities space wakes up the tutor to tell the student a potentially surprising fact: There is only one possible theory left for Card B even though there isn't a single block in AREA 2 and only two blocks have been placed! This is an example of the type of monitoring humans are not very good at.

***IS CARD B LARGE?

Student forms a conjecture. In the non-computerized version of the game, this question had only two answers; yes and no. That is, the conjecture is the same as the card or it is not. The tutor allows this to be answered in three ways: Yes, No, or Maybe. In this case LARGE is the only <u>possible</u> value Card B could have; so the answer is yes.

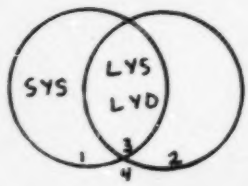
YES, THAT IS CORRECT.

***IS CARD A SMALL?

Student forms another conjecture. In this case, the tutor finds a counter-example to his conjecture so the answer is "no and here's why".

NO. IF CARD A WERE SMALL, THEN THE LARGE YELLOW SQUARE
WHICH IS IN AREA 3 WOULD BE OUTSIDE OF LOOP A.

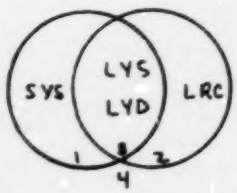
***LARGE Y DI
PUT THE LARGE YELLOW DIAMOND IN AREA 3.



***IS CARD A YELLOW?

The student forms another conjecture. In this case the tutor informs him that his "guess" is consistent with the known information but that it is just a guess because there are still other possibilities. This is an example of "maybe." Note that a human teacher would have a hard time distinguishing this fact and would simply answer yes or no depending on whether Card A was Yellow!

IT COULD BE, BUT IT COULD ALSO BE NOT RED, NOT GREEN,
NOT BLUE, NOT CIRCLE, OR NOT TRIANGLE.

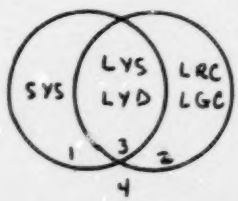


***L R CI
PUT THE LARGE RED CIRCLE IN AREA 2.

This particular block performs a perfect split of the hypothesis space, i.e. it rules out NOT GREEN, NOT BLUE, and NOT TRIANGLE.

***L G CI
PUT THE LARGE GREEN CIRCLE IN AREA 2.

Rules out NOT RED.



***S Y DI

Monitor decides that the answer to that question or block placement was deducible from the known information. Since the block the student chose contained both possible values for CARD A (it is both YELLOW and

NOT CIRCLE), it necessarily had to fall in CARD A's area (AREA 1 since CARD B is LARGE), and therefore couldn't give him any new information about the value of Card A. Since the student is placing a redundant block, the tutor is called. The tutor tries to get the student to realize that he could predict where that block will fall before it is placed.

Where do you think the SMALL YELLOW DIAMOND will go?
>>>AREA 1
It's true. It could go there.
Is there anyplace else the SMALL YELLOW DIAMOND could go?
>>>AREA 4

If the student thinks there is someplace else it could go, he must have an incorrect idea of what Card A could be (since he already knows what Card B is.)

What value could Card A have which would cause
the SMALL YELLOW DIAMOND to go in Area 4?
>>>NOT CIRCLE
If Card A were NOT CIRCLE, the SMALL YELLOW DIAMOND
would go INSIDE of Loop A and NOT in Area 4.

What value could Card A have which would cause
the SMALL YELLOW DIAMOND to go in Area 4?
>>>NOT DIAMOND
If Card A were NOT DIAMOND then the LARGE YELLOW DIAMOND
which is in Area 3 would be OUTSIDE of Loop A

What value could Card A have which would cause
the SMALL YELLOW DIAMOND to go in Area 4?
>>>NONE
Is there anyplace besides Area 1 that the SMALL
YELLOW DIAMOND could go?
>>>NO
When there is only one place a block can go, it's
not helping you guess what the cards are.

PUT THE SMALL YELLOW DIAMOND IN AREA 1.

An alternative strategy of the tutor when called by the monitor is to make the student aware that his block is not splitting his hypothesis space. The following is an example of this. The remainder of the protocol resumes after the next solid line.

---CHOOSE A BLOCK---
***S Y DI

The first step in the strategy is to get the student to verbalize the possible theories for the card. If he were wrong on any of them, counter-examples would be given. If there were more possibilities than the student lists, the tutor would work with the subset the student gives.

Can you tell me some possibilities for Card A?
>>>YELLOW AND NOT CIRCLE

Both of those are possibilities.

Next the tutor asks if his block splits his hypothesis space. Since he believes it would, the tutor walks him through the deductions to show him differently.

Will the SMALL YELLOW DIAMOND eliminate either of those theories?
>>>YES

would the SMALL YELLOW DIAMOND go INSIDE or OUTSIDE
of a loop which had a card value YELLOW?
>>>INSIDE
Right
Would the SMALL YELLOW DIAMOND go INSIDE or OUTSIDE
of a loop which had a card value NOT CIRCLE?
>>>INSIDE
Right
So you know that the SMALL YELLOW DIAMOND goes INSIDE of
Loop A no matter which of those values Card A has.

PUT THE SMALL YELLOW DIAMOND IN AREA 1.

-----CHOOSE A BLOCK----
***L Y CI
PUT THE LARGE YELLOW CIRCLE IN AREA 3.
YOU SHOULD BE APLE TO DEDUCE CARD A NOW.

This block contains the negation of one of the hypotheses for CARD A (NEGATION(NOT CIRCLE))=CIRCLE; and holds the other one constant (YELLOW). This is the correct strategy for breaking down a hypothesis space. The tutor tells him that he did just that and should now be able to deduce the card.

***IS CARD A YELLOW
YES, THAT IS CORRECT.
YOU HAVE NOW GUESSED BOTH CARDS.

--which he does!

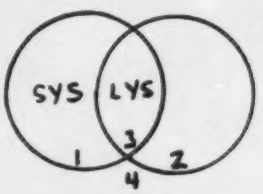
PROTOCOL 2

In this protocol, the tutor is alerted that a card can be deduced and it engages in one of its Socratic dialogues to get the student to see why this is the case. Note, these are the same two block choices that began Protocol 1.

-----CHOOSE A BLOCK----
***LARGE YELLOW SQUARE
PUT THE LARGE YELLOW SQUARE IN AREA 3.

***S Y SQ
PUT THE SMALL YELLOW SQUARE IN AREA 1.

The student should be able to figure out what Card B is. But instead of guessing it, he places another block.



***L Y DI
Can you tell me some possibilities for Card B?
>>>LARGE, SQUARE AND YELLOW

The tutor asks the student about Card B. It is possible that the student has just not considered it because the blocks both fall in loop A. When the student gives three possibilities, the tutor evaluates each giving counter-examples to the ones which are incorrect.

Card B could be LARGE.
If Card B were SQUARE then the SMALL YELLOW SQUARE
which is in Area 1 would be INSIDE of Loop B.
And if Card B were YELLOW then the SMALL YELLOW SQUARE
which is in Area 1 would be INSIDE of Loop B.

By taking the initiative, the tutor has gotten the student to observe that his reasoning is incorrect.

PUT THE LARGE YELLOW DIAMOND IN AREA 3.

The third protocol reflects a substantially expanded set of monitors. In this version, there are monitors that watch for instances when higher-order information can be logically deduced (besides just the identity of a card). Such an example might be that, given the current placement of blocks, one can logically conclude that a card could not be any SHAPE. These more abstract characterizations of the remaining possibilities for a card stem from knowing general rules or principles about the structure of this environment. For example, one rule about shapes is that if there are two blocks of the same shape, one of them inside a loop and the other outside of a loop, then the card for that loop can't be either a positive or negative shape. This is a non-trivial abstraction, and helping students to discover such abstractions is a worthwhile goal. The third protocol demonstrates monitors for these abstractions.

In addition to these monitors, the expert's capabilities have been expanded to compute a measure of the expected and actual information gain from any particular question (i.e. block choice). This information (together with the ability to remove blocks, which is included in BLOCKS though not shown) enables the student to experiment with the effectiveness of different decision strategies and enables the tutor to provide substantive hints as to what is the best next question to ask (i.e. block to choose). This facility, which would be nearly impossible to realize without a computer, introduces the student to an exceptionally powerful idea -- namely the concept of actual vs. expected information -- thus providing another point of view on the meaning of a redundant question.

PROTOCOL 3

This protocol is the same game that was played in Protocol 1, i.e. the same values for the cards and the same sequence of blocks. In this protocol, however, the information monitor and the event monitors have been turned on. The protocol also illustrates the use of the HINT command.

```
TWO CARDS HAVE BEEN CHOSEN.

---CHOOSE A BLOCK---
***LARGE Y SQ
PUT THE LARGE YELLOW SQUARE IN AREA 3.
The expected information content of that block was 1.0
The actual information content was 1.0
```

The measure of information has been normalized so that a perfect split of the hypothesis space of both cards is 1.0. The best block generally has expected information of about 1.0. (It is sometimes less than 1 because the initial set of theories is not a power of 2 so theory sets of odd number arise.) BLOCKS can print out the expected information gain of the best block as will be shown in the next interaction.

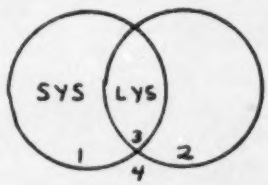
```
***SMALL YELLOW SQUARE
PUT THE SMALL YELLOW SQUARE IN AREA 1.
The expected information content of that block was .50
The actual information content was 1.67
The best block had an expected information of .99
```

Notice that in this case the student gambled with a "bad" block and won. That is, he placed a block with a low expected information but which had a very high actual information. The information monitor provides the student a new tool with which to compare strategies and choose between alternative ways of making decisions.

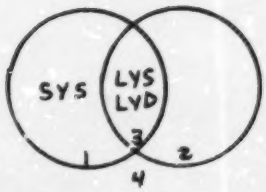
Below are the event monitors which were fired by the placement of this block. In future systems, these monitors would invoke the tutor similar to the way the placement of a redundant block does now. Notice the number of event monitors which fired -- this confirms the fact that the actual information gained by this block was quite high.

```
Card B must be size
Card B must be positive
Card B can't be shape
Card B can't be color
Card B can't be negative
Card B can't be positive shape
Card B can't be positive color
Card A can't be size
Card B can't be negative color
Card B can't be negative shape

***L Y DI
PUT THE LARGE YELLOW DIAMOND IN AREA 3.
```



The expected information content of that block was .41
The actual information content was .21

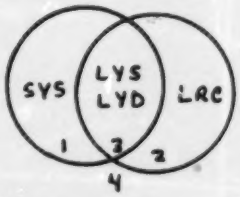


Card A can't be positive shape

***HINT
I think placing a NOT YELLOW, CIRCLE or TRIANGLE would be a good idea.

The HINT command uses the information measure to determine the set of blocks with the highest expected gain. An intensional description of this set is then calculated and given to the student. The intensional description allows the student to see which attributes are critical to the splitting of the hypothesis space.

***L R CI
PUT THE LARGE RED CIRCLE IN AREA 2.
The expected information content of that block was .5
The actual information content was .5



***HINT
I think placing a NOT YELLOW thing which is not CIRCLE, a BLUE or GREEN thing OR a NOT RED CIRCLE would be a good idea.

The possibilities for Card A are YELLOW, NOT RED, and NOT CIRCLE. These are the blocks that will split this hypothesis space.

It must now be clear that the logical information being computed for the automated tutor is very large and if the tutor were simply to pass it on to the student or, for example, query him each time a monitor deduced some fact, the student would collapse in a state of information overload. The tutor is in need of some guiding principles for determining which of this information is important to a particular student at a given moment.

Rules in the Attribute Blocks Domain

Experienced students of attribute blocks begin to notice interesting relationships among the placed blocks that can lead to conclusions that are not immediately obvious. For example, if there are red blocks both inside and outside Loop A, then Card A cannot be any positive or negative color! (Proof of this theorem is left as an exercise to the reader.) An expert student of attribute blocks recognizes and uses such relationships to guide

his hypothesis formation. The computerized expert in an intelligent instructional system must be able to do the same. A central premise of intelligent knowledge-based CAI is that good tutoring can point out structure in an environment which might have otherwise been missed: and in so doing, it allows the student to enrich his understanding of (and skills in) the environment. This statement is particularly true in the attribute blocks domain, where much structure exists, and much can be overlooked without the assistance of a tutor.

The attribute blocks domain, much like the domain of geometry, can be characterized by a set of axioms and theorems. The basic axioms of the Blocks world are extremely simple: Each block has exactly three attributes whose values are selected from one of the three lists, no two blocks have more than two attribute values in common, etc. As in geometry, it is the theorems that provide an interesting description of the domain. Although the Blocks domain can be described in a formal mathematical manner, that is not the purpose of enumerating its theorems: the interest here is in providing a basis for grasping some of the principles of deduction and decision making.

In attribute blocks, as in geometry, "higher-level" theorems can be built upon "lower-level" ones; that is, there exists a minimal set of rules from which all others can be derived. In the attribute blocks world, this minimal set consists of just one theorem (also viewable as a pair of theorems), along with, of course, the axioms that define the game. (This theorem is the first in the list below.) By providing BLOCKS with more than this one theorem, the system reflects the ability of people to think in higher-level terms.

The set of four theorems given below represents an effort to enumerate those theorems which are: in some way "basic" to the attribute blocks domain; easily provable; most likely to be used by a person playing attribute blocks; and a good basis for a set of tutoring strategies for the game. Note that this set is neither minimal nor exhaustive. The four theorems are:

Given a Loop L and associated card C,

1. For any block inside L, rule out all positive attribute values except those exhibited by that block, and all negatives of the values exhibited by the block, from the possibilities for C. For any block outside L, rule out all positive attributes of the block and all negatives of attributes not exhibited by the block.
2. If the (positive) value of an attribute appears on blocks both inside and outside L, then rule out all values, positive and negative, of that attribute for C.
3. If all values of an attribute appear either inside L or outside L, then rule out all values of that attribute from the possibilities for C.
4. If 2 or more, but not all, (positive) values of an attribute appear inside L, then rule out all positive values of that attribute. If 2 or more values appear outside L, then rule out all negative values of that attribute.

A fifth rule, which may seem obvious but is nonetheless important, is that if all subcategories of an attribute have been eliminated, then rule out that attribute.

Rule Application

The abstractions of values into higher-order attributes can be viewed as a "theory" tree such as is shown in Figure 3. The process of deducing a card in attribute blocks is equivalent to pruning the theory "tree" for that card until all but one path from the top node to the terminal attribute value is eliminated. The expert maintains this tree by examining the environment (configuration of placed blocks) and determining which rules are applicable.(1) When it finds an applicable rule, it "kills" the tree node(s) corresponding to the eliminated possibilities by attaching information to them on why they were killed. This information includes the name of the rule that was invoked and the names of the blocks that caused the rule to succeed. Thus, the "live" nodes are those that have no such information attached.

(1) One fact of the blocks world is that even though the two loops intersect, their theory trees are independent, so the expert processes one loop at a time.

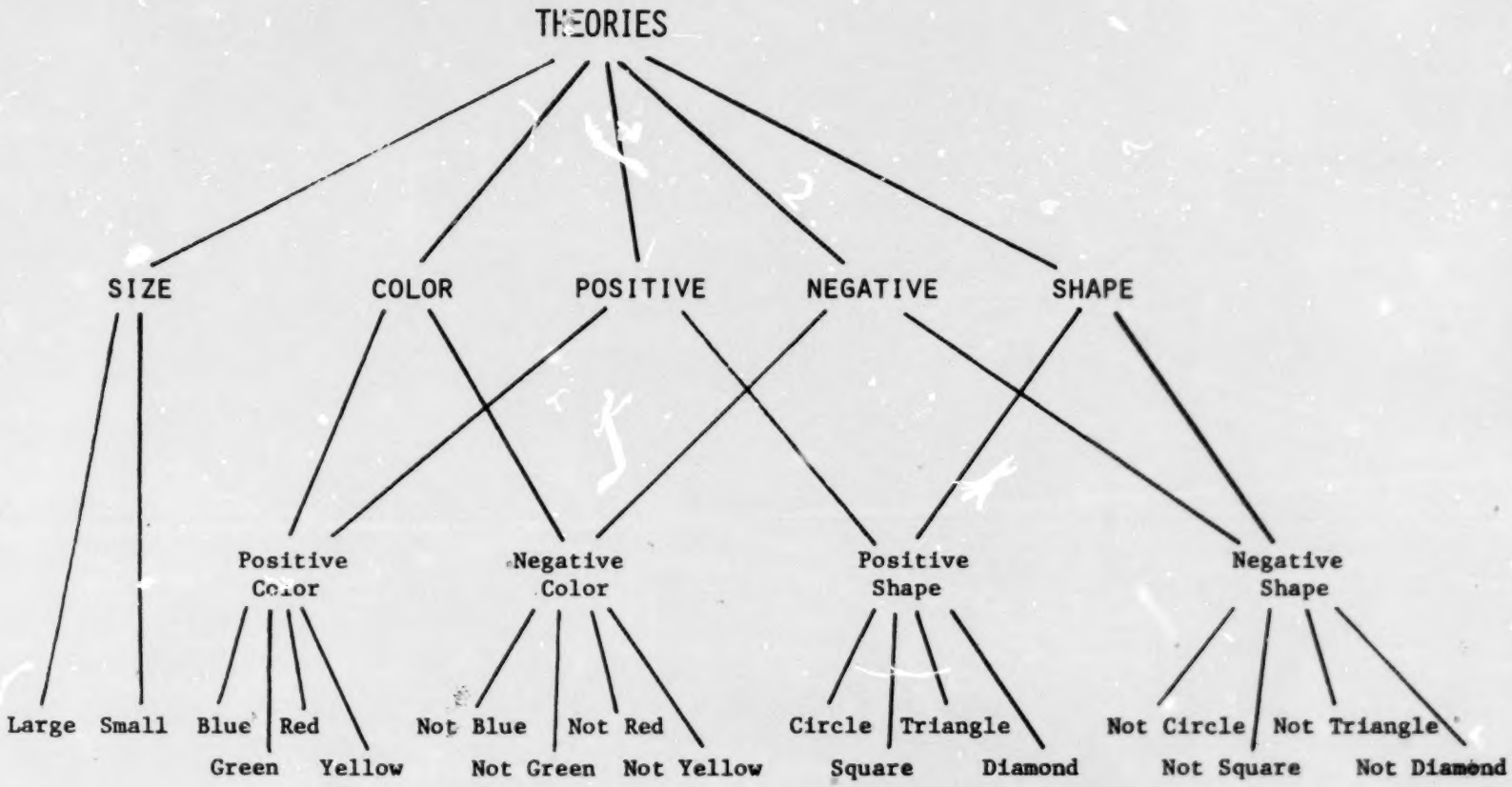

THEORIES
SIZE
COLOR
POSITIVE
NEGATIVE
SHAPE
Positive Color
Negative Color
Positive Shape
Negative Shape
Large
Small
Blue
Red
Green
Yellow
Not Blue
Not Red
Not Green
Not Yellow
Circle
Triangle
Square
Diamond
Not Circle
Not Triangle
Not Square
Not Diamond

Figure 3

The augmented theory trees maintained by the rule-based expert constitute a large data base for the tutor. By examining the trees, the tutor can answer not only questions like "Is Card A green?", but also those like "Can Card A be a negative color?", "Card A must be positive, right?", and "Why can't Card A be a shape?", and provide reasons for its answers that the student can comprehend in terms of the theorems. Even with this data base, however, the problem of how to answer the general question "Why can't card x be y?" is complex. For example, suppose that the tutor is called upon to explain why Card B cannot have the value RED. The tutor then examines the theory tree for Card B and discovers that the nodes POSITIVE COLOR, POSITIVE, and COLOR have all been killed, as well as the RED node. Does the tutor then explain that Card B can't be RED because it can't be any color, because it can't be positive, because it can't be a positive color, or just because RED itself is contradicted?

The solution lies in trying to understand the reasoning process used by the student. If the student, through his block placement and/or questions to the tutor, has shown a good grasp of the theorems involved in the deductions necessary to rule out RED, then it is likely that he has merely overlooked a placed block that contradicts his hypothesis.(2) On the other hand, if the student has failed to exhibit understanding of theorems, the tutor begins its explanation at the lowest level theorem that the student has failed to grasp. This, of course, begs the difficult question of how the tutor can determine what the student knows. This is a continuing research area.

System Description

The BLOCKS system has been structured to allow experimentation with various tutorial and assistance modules. For more detail, the reader is directed to [Brown et al. 1975]. In this section we will describe the overall structure of BLOCKS, and the workings and responsibilities of the individual modules. Figure 2 shows the basic organization of processes and data within the Blocks system.

(2) Having overlooked a block is not a possibility if the theory tree shows more than one way of ruling out RED which use different blocks.

Executive

The executive has responsibility for the control flow within the system. The typical control path to process a student's statement is as follows: The executive reads the input statement from the student and passes it to the natural language understander. The natural language understander identifies the intent of the statement and returns its semantic structure which contains the pertinent information for one of the environment maintainers (unless, of course, the statement was directed to the tutor or one of the monitors). The executive calls the proper maintenance routine which carries out the appropriate change to the environment. At this point the tutor knows the student's move and what the result of that move will be. However, before it tells the student where the block goes, it can query the student about what he has in mind by placing that block or he can point out things that the student should have known but didn't (because if he had, his present block choice would have been different). After the tutor has finished, the executive calls the natural language generator to tell the student where the block goes. Then the tutor is called (again invoking the monitors) to decide whether to further explain the results of the block choice or the ramifications of it.

Natural Language Understander

The classes of sentences used in BLOCKS to date have been fairly straightforward and are handled quickly and easily by a small semantic grammar processor [Burton 1976]. The Blocks grammar has about 15 semantic categories with very little complexity. The flexible framework provided by the semantic grammar was particularly useful in writing rules to recognize descriptions of blocks. Without it, the understander would have been much harder to write.

Environment Maintenance

BLOCKS is designed around the environment of a student playing with attribute blocks. This environment consists of the values of the cards, the locations of blocks which have been placed, and the possible theories for the cards which are consistent with the placed blocks.(3) The tasks

(3) The list of possibilities can be recalculated from the blocks but was deemed important enough to make it part of the environment.

involved in maintaining the blocks environment are performed by procedural specialists called environment maintainers. These tasks include placing and removing blocks (including determining where blocks should go), printing the present board configuration, setting up the cards to begin a game and stopping the session. The effect of each of these maintenance actions is to change some portion of the data base which is examined at various times by the other modules.

Monitors

To study the effects of various types of services which the attribute blocks laboratory could provide, various monitors were designed and implemented. We shall describe two: the Remaining Possibilities monitor and the Information Gain monitor.

Remaining possibilities monitor:

In order to allow the student to see the effect that placing certain blocks had on the theory sets for each of the cards, a monitor was written which calculates all of the possible values for each of the cards from a configuration of blocks. By invoking this monitor via a call to HELP, together with being able to remove blocks, the student can discover how certain blocks (questions) will split a set of possible theories. This monitor can also inform the tutoring routines when redundant blocks are placed or when a set of possible theories is reduced to one element.

Information Gain monitor:

The attribute blocks world is an excellent domain to study problems of decision-making such as what makes a good question. The expected information gain of a block and its actual information gain provide a valuable metric for evaluating alternative questions. The expected information gain of a block is the sum over the four areas of the amount of information gained by that block falling in that area, times the probability of it falling there. (The probability of a block falling in an area depends upon the remaining possibilities of a card.) The actual amount of information gained from a block falling in an area is the logarithm of the percentage of possible theories that block eliminated (in the cross product space of theories for card A and theories for card B).

The logarithm is taken base 4 since each question has four possible answers (each block could go in one of four possible areas). Since the beginning theory space has 324 members (18x18), the expected number of questions required to isolate one individual element is LOG 324 (base 4) or about 4.2. When the total actual information gain of the student's blocks totals 4.2, he can deduce both cards. By seeing both the expected and actual information gain, the student can begin to develop intuitions about "good" questions.

The Tutor and Its Tutoring Strategies

The tutor is invoked when either: 1) the student has placed a redundant block; or 2) the student has chosen another block when he should have been able to deduce a card from the currently placed blocks. The Tutor attempts through a series of Socratic dialogues to direct the student's attention to aspects of the situation that he may have missed. Protocols 1 and 2 show several examples of the Tutor intervening in a student's session. At present the Tutor has three Socratic strategies:

1) If the student fails to deduce a card, try to get him to say what he thinks it could be and show him by counter-example where he is wrong;
2) If the student places a redundant block, try to get him to predict where it will go and convince him that is the only place it could go;
3) If the student places a redundant block, get him to articulate some remaining theories and to choose a block which would distinguish between them.

From experiments, we have found this tutoring to be valuable although at present it is much too oppressive! When to tutor and when not to is a very difficult problem whose solution will require a structural model of the student together with a better understanding of the effect of various teaching strategies. One of the purposes of BLOCKS is to provide a good framework in which to explore these issues.

Expert Rule-based Deducer

Since the basic attribute blocks rules described earlier represent a set of discrete production rules, the decision was made to use an existing production rule interpreter for their implementation. The Rand Intelligent Terminal Agent (RITA), which was developed at Rand and augmented and implemented in INTERLISP by R. Bobrow, was chosen. RITA accepts rule sets

in an English-like notation and translates them into LISP functions which may then be compiled and applied much like any other LISP function.

In (the LISP version of) RITA, objects are represented as list structure, in property-list format. The property names are the objects' attributes; and the property values are the values of those attributes possessed by the objects. For example, the representation of the 32 attribute blocks is a list of 32 elements, each of the form:

```
(NAME LRS SIZE LARGE COLOR RED...)
```

One of the primary features of RITA is the ability to "instantiate" objects based on their attributes. One can also call other LISP functions, whether hand-coded or compiled from RITA rule sets, from inside a RITA rule set. (4)

An example will serve to illustrate the format of RITA rules. One rule in the main rule set of the rule-based expert eliminates the node COLOR from the theory tree of a loop if the same color is represented both inside and outside the loop. In slightly abridged form, that rule looks like this:

```
rule COLOR/IN&OUT
     if    there is a BLOCK(B1) whose name is in CONTENTS of LOOP
           and there is a BLOCK(B2) whose NAME is in OUTSIDE of
           LOOP
            and whose COLOR is COLOR of BLOCK(B1)
     then  SAYCAN'TBE(COLOR, NAME OF LOOP, COLOR/IN&OUT,
                      BLOCK(B1), BLOCK(B2)) ;
```

In this example, lowercase words are RITA keywords and uppercase words are names of objects and attributes in the RITA environment. LOOP is a parameter handed to the rule set, and has attributes that indicate which blocks have been placed inside and outside of it. SAYCAN'TBE is a LISP function responsible for massaging the theory trees. In the actual rule set used by the the expert, the rules are slightly more complicated to eliminate duplications and permutations.

Conclusion

In this paper, we have described the philosophy of intelligent instructional systems and presented an example of such a system (BLOCKS). The notion of BLOCKS as a paradigmatic system was explicated from both the

(4) See the RITA User's Manual [Weissman & Bobrow 1976] for a description of the operation and capabilities of RITA.

system development and educational points of view. From a developmental point of view, the modular design of BLOCKS provides a functioning framework within which to explore different monitoring functions and various tutoring strategies. A particularly interesting discovery that arose from experimenting with the system was the need for a structural model of the student to direct the content and level of the tutor's comments. From an educational viewpoint, BLOCKS provides a dramatic example of the potential of a computerized intelligent tutor in a laboratory environment. By monitoring the student's behavior, the system can notice interesting events and direct the student's attention to them. In this way, the computer can actually provide conceptual structure and guidance to a student's otherwise undirected experiences.

Acknowledgements

We are especially indebted to Dr. Guy Groen whose suggestion led us to explore the use of production rules as a representation for student modelling and who coined the term "paradigmatic system". We would also like to express our thanks to Robert Weissman for developing the RITA rules for BLOCKS, to Kathy Larkin for her suggestions and assistance on this paper, and to Mark Spikell who suggested we consider the domain of attribute blocks for developing our intelligent tutoring paradigms.

This research was supported by the Advanced Research Projects Agency, Air Force Human Resources Laboratory, Army Research Institute for Behavioral and Social Sciences, and Navy Personnel Research and Development Center, under Contract No. MDA903-76-C-0108.

References

Brown, J.S. "Uses of Artificial Intelligence and Advanced Computer Technology in Education" Proceedings of the conference Ten-Year Forecast for Computers and Communication: Implications for Education: 1975.

Brown, J.S., Burton, R.R. et al. "Steps Toward a Theoretical Foundation for Complex Knowledge-Based CAI" BBN Report No. 3135, ICAI Report No. 2, Bolt Beranek and Newman Inc., Cambridge: 1975. (See Chapter 5.)

Brown, J.S., Burton R.R., Bell A.G. "SOPHIE: A Sophisticated Instructional Environment for Teaching Electronic Troubleshooting (An Example of AI in CAI)" International Journal of Man Machine Studies 7 1975.

Brown, J.S., Burton, R.R. et al. "Aspects of a Theory for Automated Student Modelling" BBN Report No. 3549, ICAI Report No. 4, Bolt Beranek and Newman Inc., Cambridge: 1977.

Brown, J.S., Rubinstein, R. and Burton, R. "Reactive Learning Environment for Computer-Assisted Electronics Instruction" BBN Report No. 3314, ICAI Report No. 1, Bolt Beranek and Newman Inc., Cambridge: 1976.

Burton, R.R. "Semantic Grammar: An Engineering Technique for Constructing Natural Language Understanding Systems" BBN Report No. 3453, ICAI Report No. 3, Bolt Beranek and Newman Inc., Cambridge: 1976.

Burton, R.R. and Brown, J.S. "A Tutoring and Student Modelling Paradigm for Gaming Environments" Proceedings for the Symposium on Computer Science and Education, Anaheim, California: 1976.

Carr, B. and Goldstein, I.P. "Overlays: A Theory of Modelling for Computer Aided Instruction" AI Memo 406, Logo Memo 40, Artificial Intelligence Laboratory, M.I.T., Cambridge: 1977.

Dienes, Z.P. and Golding E.W. Learning Logic, Logical Games Herder and Herder, New York: 1970.

Elementary Science Study Attribute Games and Problems McGraw-Hill Book Company, New York: 1968.

Goldstein, I.P. "A Preliminary Proposal for Research on the Computer as Coach: An Athletic Paradigm for Intellectual Education" AI Memo 389, Logo Memo 37, Artificial Intelligence Laboratory, M.I.T., Cambridge: 1976.

Greenes, C.E., Willcutt, R.E. and Spikell, M.A. Problem Solving in the Mathematics Laboratory Prindle, Weber & Schmidt, Inc. Boston: 1972.

Weissman, R.L. and Bobrow, R.J. "RITA User's Manual" internal memo, Bolt Beranek and Newman Inc., Cambridge: 1976.